Plugin Monitoring for GLPI

Contents

Preface

Introduction

Introduction

But de ce projet

Description des buts de ce projet

Le but de ce plugin est de :

- Piloter la configuration de Shinken monitoring via GLPI
- Récupérer l'état des services en temps réel dans GLPI
- Avoir l'historique des contrôles avec les graphiques RRDTOOL
- Créer des tickets d'assistance dans le helpdesk de GLPI lors du'un évènement (planifié)
- Utiliser les fonctionnalités de Shinken comme les règles métier...

Pourquoi est-ce révolutionnaire?

Une révolution?

Le plugin Monitoring introduit de nouveaux concepts dans les systèmes de monitoring :

- Fini le temps où on ajoutait chaque service à un équipement (ordinateurs, serveurs, switch...) et/ou ajoutait dans des templates... **On se sert désormais de l'INVENTAIRE des équipements pour définir automatiquement les ressources à surveiller** (donc sans intervention humaine, excepté pour la configuration au début quant on installe le plugin Monitoring). Cette fonctionnalité peut être utilisée dans le cas d'un système d'inventaire automatique tel que FusionInventory.

  

  Note: Par exemple, tous les ordinateurs avec un nom commençant par "SRV" et qui ont Apache d'installé, vont avoir un check_http et check_apachestatus.

- On a créé des gabarits standard pour les RRDTOOL au format JSON et ils peuvent être utilisés par d'autres systèmes de monitoring. 2 types de gabarits ont été créés :
  - to convert perf_data of checks to RRDTOOL files
  - for graph generation based on these RRDTOOL files previously created and update with data

Liens

Comment contacter le projet?

Site internet

Site internet du projet

Vous pouvez trouver les sites internet des projets à l'adresse :

- *https://forge.indepnet.net/projects/monitoring/*

Les gabarits RRDTOOL sont situés à cette adresse :

- *https://github.com/rrdtooltemplates/rrdtool_templates*

Canal IRC

Adresse du canal IRC

Voivi les informations des canaux IRC des projets :

- Serveur: *freenode.net*
- Cannal : *#glpimonitoring*
- Demander l'utilisateur *ddurieux*

Termes utilisés dans cette solution

Certains termes sont mieux définis

Nous avons redéfinis les termes pour qu'ils soient plus faciles à utiliser.

Les termes de monitoring sont en italiques et les termes du plugin Monitoring sont en gras :

- *Services templates* => **Composants**
- *Services* => **Ressources**
- *Timeperiods* => **Calendrier**
- *Business Rules* => **Catalogues de services**

Part
I

Préparation

Topics:

Les éléments nécessaires pour préparer la solution

Ce chapitre détaille la préparation de l'installation et son utilisation

Outils indispensables

Voici la liste et la description des outils indispensables

Voici la liste des outils indispensables pour cette solution :

GLPI (Gestion Libre de Parc Informatique)

GLPI est un outil open source for la gestion d'actifs et d'assistance helpdesk

Licence: GPLv2

Site internet: *http://www.glpi-project.org/*

Version requise : 0.80.x (si possible la dernière version disponible : 0.80.7)

Il y a plusieurs plugins disponibles qui étendent les fonctionnalités.

FusionInventory

FusionInventory est un outil qui s'occupe de l'inventaire des ordinateurs, serveurs, switchs, imprimantes réseau...

Site internet: *http://www.fusioninventory.org/*

FusionInventory se compose de 2 parties, un agent :

Licence: GPLv2

Version requise: 2.1.x (si possible la dernière version disponible : 2.1.14)

et un plugin pour GLPI (côté serveur)

Licence : AGPLv3

Version requise : 0.80+1.1

Plugin Webservices pour GLPI

C'est un plugin pour GLPI qui ajoute l'accès à GLPI par service web (XML-RPC et SOAP)

Licence: GPLv2

Site internet: *https://forge.indepnet.net/projects/webservices*

Version requise : 1.2.0

Shinken

Shinken est un outil de monitoring (fork de Nagios) écrit en Pyhton. Shinken va faire les crontrôles de ressources.

Licence : AGPLv3

Site internet: *http://www.shinken-monitoring.org/*

Version requise : 1.0.1

Plugin Monitoring pour GLPI

C'est ce plugin qui va piloter Shinken à partir de GLPI (et pour d'autres système de monitoring plus tard)

Licence : AGPLv3

Site internet: *https://forge.indepnet.net/projects/monitoring/*

Version requise : 0.80+1.2

Application Monitoring for GLPI pour Android

Comment cette solution fonctionne

Pour mieux appréhender l'utilisation, voici une description du fonctionnement de la solution

Légende :

- *[D]* = opération automatique
- *[M]* = opération manuelle
- *[DM]* = peut être fait dynamiquement ou manuellement

Voici la description des processus :

- *[M]* Un technicien installe un nouveau serveur
- *[DM]* Installation de l'agent FusionInventory pour avoir un inventaire complet de ce serveur
- *[D]* FusionInventory envoi un inventaire à GLPI
- *[D]* le plugin Monitoring défini les ressources pour ce nouveau serveur en utilisant les règles (dans le catalogue de composants)
- *[DM]* Redémarrer le module Arbiter de Shinken afin de charger la nouvelle configuration
- *[D]* Shinken s'exécute avec les nouveaux contrôles
- *[D]* Shinken envoi les évènements dans la base de données MySQL de GLPI
- *[D]* Une tâche système planifiée met à jour les fichiers RRDTOOL avec des évènements et génère / met à jour les graphiques
- L'utilisateur peut voir les évènements dans GLPI ou avec une application sur son smartphone Android

Conclusion : Ajouter un nouvel équipement à monitorer est simple et ne requiert pas d'intervention humaine.

Part
II

Installation

Topics:

Installation de tous les élements/outils

Suivre ce tutorial pour avoir la solution fonctionnelle :

Shinken

Installation de Shinken

Pour une installation simple en 10 minutes de Shinken, voir la documentation officielle à l'adresse :*http://www.shinken-monitoring.org/wiki/shinken_10min_start*

Shinken requiert une version 2.6 or 2.7 de Python.

GLPI

Installation de GLPI

Pour l'installation de GLPI dans votre environnement, se reporter à la documentation officielle à l'adresse : h*ttps://forge.indepnet.net/projects/glpidoc/files*

FusionInventory

Installation de FusionInventory

Il faut installer et configurer l'environnement FusionInventory (serveur et agent). Pour celà, se reporter à la documentation de FusionInventory disponible à l'adresse : *http://fusioninventory.org/wordpress/documentation/*

Plugin Webservices pour GLPI

Installation du plugin Webservices

- Récupérer la dernière version disponible de GLPI : *https://forge.indepnet.net/projects/webservices/files*
- Décompresser l'archive téléchargée dans <glpiFolder>/plugins/
- Se connecter au serveur GLPI (web)
- Aller dans *Configuration > Plugins*
- Cliquer sur *installer* pour intégrer le plugin Webservices à GLPI
- Cliquer sur *activer* pour activer le plugin WebServices
- Le plugin est installé

Plugin Monitoring pour GLPI

installation du plugin Monitoring

- Récupérer la dernière version disponible de GLPI : *https://forge.indepnet.net/projects/monitoring/files*
- Décompresser l'archive téléchargée dans <glpiFolder>/plugins/
- Se connecter au serveur GLPI (web)
- Aller dans *Configuration > Plugins*
- Cliquer sur *installer* pour intégrer le plugin Monitoring dans GLPI
- Cliquer sur *activer* pour activer le plugin Monitoring
- Le plugin est installé

Android : application glpi_monitoring (optionnel)

Installation de l'application glpi_monitoring pour Android

- Téléchargez le fichier APK avec le navigateur d'Android : *https://forge.indepnet.net/projects/monitoring/files*
- Ouvrir le package APK et installer l'application.

- Cette application est installée

Part
III

Configuration

Topics:

Configuration des éléments/outils

Ce chapitre décrit comment configurer les composants

Gestion des ETIQUETTES

Gestion des ETIQUETTES

Important: Dans la plupart des cas, l'utilisation des ETIQUETTES n'est pas requise, donc il vaut mieux laisser cette configuration vide.

Dans certains cas, on veut avoir un serveur Shinken associé à une entité GLPI (donc avoir 2 serveurs Shinken associés à 2 entités différentes). Cette configuration est utilisée quand chaque entité est un site différent et distant et n'a pas de liens entre eux, mais ils ont un lien possible avec GLPI (hébergé sur un serveur web par exemple). Donc dans ce cas, il faut définir une étiquette à une entité de GLPI et au serveur Shinken.

- Dans Shinken, c'est dans la configuration du module Arbiter nommé *GLPI*.
- Dans GLPI, c'est dans l'onglet *Monitoring* dans le formulaire des entités.

Création des comptes GLPI

Création de compte GLPI

Il faut créer des comptes GLPI :

- Shinken a besoin d'un compte GLPI pour se connecter au plugin Webservices afin de récupérer sa configuration. Créer un compte local GLPI via *Administration* > *Utilisateur*.

  Par exemple :

  - Login: *shinken*
  - Password: *passshinken*
- Compte(s) d'utilisateur normal : ne pas utiliser le compte super-admin de GLPI, il faut créer votre propre compte utilisateur (ou importer du LDAP, voir la documentation de GLPI pour ce cas)

Webservice de GLPI

Configurer le webservice de GLPI

Pour Shinken

Configurer les services web de GLPI pour Shinken

Dans GLPI, aller dans le menu *Plugins* > *Web Services*, et ajouter un nouveau service web. Mettre les valeurs :

- Nom : *Shinken*
- Services actifs : *Oui*
- Activer la compression : *Non* (pas testé avec la compression activée)
- Tracer les connexions : *Non* (activez le si vous voulez garder une trace des connexions)
- Debug : *Non* (activer pour le debugging)
- Motif SQL des services : .*
- Plage d'adressage IP : définir une plage IP, dans ce cas, mettre 2 fois la même IP du serveur ou Shinken est installé
- Utilisateur: (laisser vide dans ce cas)
- Mot de passe: (laisser vide dans ce cas)

Cliquer sur le bouton ajouter. Le service web pour Shinken est bien créé et donne accès à Shinken.

Pour l'application Android

Configurer le webservice de GLPI pour l'application Android

Configurer uniquement si vous souhaitez utiliser l'application Android

Dans GLPI, aller dans le menu *Plugins > Web Services*, et ajouter un nouveau service web. Mettre les valeurs :

- Nom : *Android glpi_monitoring*
- Services actifs : *Oui*
- Activer la compression : *Non* (pas testé avec la compression activée)
- Tracer les connexions : *Non* (activez le si vous voulez garder une trace des connexions)
- Debug : *Non* (activer pour le debugging)
- Motif SQL des services : *monitoring.**
- Plage d'adresses IP : si vous avez une IP fixe sur votre smartphone Android (rarement le cas), définissez cette IP dans les 2 cases, autrement vous aller mettre une très grosse plage comme *1.0.0.1* à *254.254.254.254*
- Utilisateur: (laisser vide dans ce cas)
- Mot de passe: (laisser vide dans ce cas)

Cliquer sur le bouton ajouter : le service web pour l'application Android est créé et permet d'accéder à GLPI.

Création des comptes MySQL

Créer un compte MySQL

Il faut créer un compte MySQL car le module *Broker* de Shinken va ajouter et mettre à jour des évènements dans la base de donées.

Par exemple créer :

- Login : *shinkenbroker*
- Password : *passshinken*
- Host : IP du serveur sur lequel Shinken est installé

You pouvez définir les droits sur toute la base GLPI mais, pour une meilleure sécurité, donnez les droits uniquement aux tables

- base glpi\glpi_plugin_monitoring_services (SELECT et UPDATE seulement)
- base glpi\glpi_plugin_monitoring_serviceevents (INSERT seulement)
- glpi database\glpi_plugin_monitoring_servicescatalogs (SELECT et UPDATE uniquement)

Shinken

Configurer Shinken

Module Arbiter

Configurer le module Arbiter de Shinken

Le module *Arbiter* est utilisé et chargé dans la configuration de Shinken. Ouvrir le fichier de configuration *shinken-specific.cfg* et modifier ce module :

```
define module{
       module_name GLPI
       module_type glpi
       uri http://localhost/glpi/plugins/webservices/xmlrpc.php
       login_name glpi
       login_password glpi
       tag
```

```
}
```

Les valeurs à modifier pour votre environnement sont :

- uri: url de GLPI, se termine toujours par */plugins/webservices/xmlrpc.php*
- login_name: compte GLPI créé dans "créer des comptes GLPI" (dans notre exemple c'est *shinken*)
- login_password: mot de passe du compte GLPI (dans notre exemple c'est *passshinken*)
- tag: défini l'étiquette si on l'utilise, sinon laisser vide

Le module est désormais configuré, il faut le rajouter dans la configuration du module Arbiter :

```
define arbiter{
       arbiter_name Arbiter-Master
# host_name node1 ;result of the hostname command under Unix
       address localhost ;IP or DNS address
       port 7770
       spare 0
# uncomment the line below if you want to use some modules for your arbiter
# modules CommandFile, Mongodb, NSCA, VMWare_auto_linking

# List of interesting modules :
# CommandFile : open the named pipe nagios.cmd
# Mongodb : load hosts from a mongodb database
# PickleRetentionArbiter : save data before exiting
# NSCA : NSCA server
# VMWare_auto_linking : lookup at Vphere server for dependencies
# GLPI : import hosts from GLPI
# TSCA : TSCA server
[...]
```

Donc il faut ajouter le module et nous allons avoir :

```
define arbiter{
       arbiter_name Arbiter-Master
# host_name node1 ;result of the hostname command under Unix
       address localhost ;IP or DNS address
       port 7770
       spare 0
# uncomment the line below if you want to use some modules for your arbiter
# modules CommandFile, Mongodb, NSCA, VMWare_auto_linking

modules GLPI

# List of interesting modules :
# CommandFile : open the named pipe nagios.cmd
# Mongodb : load hosts from a mongodb database
# PickleRetentionArbiter : save data before exiting
# NSCA : NSCA server
# VMWare_auto_linking : lookup at Vphere server for dependencies
# GLPI : import hosts from GLPI
# TSCA : TSCA server
[...]
```

Module Broker

Configurer le module broker de Shinken

Le module Broker est utilisé pour envoyer des évènements. Ouvrir le fichier de configuration *shinken-specific.cfg* et modifier ce module :

```
# send into GLPI DB
```

```
# =============== Work with Plugin Monitoring of GLPI ===============
define module{
  module_name glpidb
  module_type glpidb
  database glpi ; database name
  user root ; database user
  password root ; must be changed
  host localhost ; host to connect to
}
```

Les valeurs à modifier sont :

- database: nom de la base MySQL de GLPI
- user: compte MySQL créé au chapitre *Création des comptes MySQL* on page 17 (dans notre exemple, c'est *shinkenbroker*)
- password: mot de passe du compte GLPI (dans notre exemple, c'est *passshinken*)
- host: IP ou nom du serveur où le serveur MySQL est installé.

Le module est défini, désormais ajoutons le dans la configuration du module Arbiter :

```
#The broker manages data export (in flat file or in database)
#with its modules
#Here just log files and status.dat file modules
define broker{
       broker_name broker-1
       address localhost
       port 7772
       spare 0

       # Which modules to load? LiveSatus and logs by default.
       modules Livestatus, Simple-log, WebUI



# Other interesting modules to add :
# PickleRetentionBroker : save data when quitting
# ToNdodb_Mysql : NDO database support
# NPCDMOD : Use the PNP addon
# Graphite-Perfdata : Use teh Graphite backend for perfdata
# WebUI : Shinken Web interface
[...]
```

Donc il faut ajouter le module et nous allons avoir :

```
#The broker manages data export (in flat file or in database)
#with its modules
#Here just log files and status.dat file modules
define broker{
       broker_name broker-1
       address localhost
       port 7772
       spare 0

       # Which modules to load? LiveSatus and logs by default.
       modules Livestatus, Simple-log, WebUI, glpidb



# Other interesting modules to add :
# PickleRetentionBroker : save data when quitting
# ToNdodb_Mysql : NDO database support
```

```
# NPCDMOD : Use the PNP addon
# Graphite-Perfdata : Use teh Graphite backend for perfdata
# WebUI : Shinken Web interface
[...]
```

Dans une implémentation simple, Livestatus, Simple-log et WebUI peuvent être supprimés au chargement à la line commençant par *modules*.

Objets communs

charge les objects communs

Par défaut, Shinken charge les object communs comme les commandes, timeperiod, hôtes, services. Si vous souhaitez charger uniquement la configuration avec GLPI, vous devez modifier le fichier *nagios.cfg* :

```
#Configuration files with common objects like commands, timeperiods,
#or templates that are used by the host/service/contacts
#cfg_file=commons.cfg
cfg_file=commands.cfg
cfg_file=timeperiods.cfg
cfg_file=escalations.cfg
cfg_file=dependencies.cfg
cfg_file=contacts.cfg
#cfg_dir=/opt/omd/sites/pilot/etc/nagios/conf.d/pilot/dynamic

#Now templates of hosts, services and contacts
cfg_file=templates.cfg

#Now groups
cfg_file=servicegroups.cfg
cfg_file=hostgroups.cfg
cfg_file=contactgroups.cfg

#and now real hosts, services, packs and
# discovered hosts
cfg_dir=hosts
cfg_dir=services
cfg_dir=packs
cfg_dir=objects/discovery

# Un comment this for a sample configuration
#cfg_dir=sample
[...]
```

Par

```
#Configuration files with common objects like commands, timeperiods,
#or templates that are used by the host/service/contacts
#cfg_file=commons.cfg
#cfg_file=commands.cfg
#cfg_file=timeperiods.cfg
#cfg_file=escalations.cfg
#cfg_file=dependencies.cfg
#cfg_file=contacts.cfg
#cfg_dir=/opt/omd/sites/pilot/etc/nagios/conf.d/pilot/dynamic

#Now templates of hosts, services and contacts
#cfg_file=templates.cfg

#Now groups
#cfg_file=servicegroups.cfg
#cfg_file=hostgroups.cfg
```

```
#cfg_file=contactgroups.cfg

#and now real hosts, services, packs and
# discovered hosts
#cfg_dir=hosts
#cfg_dir=services
#cfg_dir=packs
#cfg_dir=objects/discovery

# Un comment this for a sample configuration
#cfg_dir=sample
[...]
```

Plugin monitoring

Configurer le plugin Monitoring

Configuration des définitions de contrôle

Configuration des contrôles

Cette définition des contrôles est utilisée pour définir une ressource, quel type de contrôle (toutes les 5 minutes avec 3 essais, toutes les 15 minutes avec 2 essais...).

Pour accéder à ce menu, aller à *Plugins* > *Monitoring* > *Définition de contrôle*.

Les élements à configurer pour chaque définition de contrôle sont :

- *Nom* : donner un nom.
- *Temps en minutes entre 2 contrôles* : défini le temps en minutes entre 2 contrôles.
- *Nombre maximum de tentatives (nombre d'essais)* : défini le nombre d'essais quand un état (OK, warning, critical...) change avant d'avoir réellement ce nouveau état.
- *Temps en minutes entre 2 essais* : temps entre 2 essais.

Configuration des commandes

Configuration des commandes

Ces commandes sont utilisées pour les contrôles et détecte l'état des services à monitorer.

Plusieurs commandes sont ajoutées lors de l'installation du plugin.

Pour accéder à ce menu, il faut aller à *Plugins* > *Monitoring* > *Commandes*

Les éléments à configurer pour chaque commande de contrôle sont :

- *Nom*: donner un nom
- *Nom de la commande* : défini le nom de la commande, il doit être unique et sans caractères spéciaux. C'est le nom du script que Shinken va exécuter pour vérifier ce que vous souhaitez (disque, cpu, mémoire...).
- *Ligne de commande* : chemin et nom de fichier à exécuter (sur le serveur où Shinken est installé puisque c'est Shinken qui va exécuter cette commande). Il est possible d'utiliser les Macro Shinken comme *$PLUGINSDIR$*, *$MYSQLUSER$*... et on peut utiliser les macro pour les arguments dynamiques utilisés dans le plugin monitoring comme *$ARG1$*, *$ARG2$*, *$ARG3$*...
- Description des arguments : ceci est visible uniquement quant on a *$ARG1$*, *$ARG2$*, *$ARG3$*... dans le champ *Ligne de commande*. Pour chaque argument, on a une case a remplir avec la description de cet argument. Par exemple, si on a dans *Ligne de commande* la commande *$PLUGINSDIR$/check_load -r -w $ARG1$ -c $ARG2$*, il va y avoir 2 cases à remplir, une pour chacun des 2 arguments et dans *ARG1* on défini à quoi cet argument sert : *l'état WARNING si le load average dépasse WLOADn (WLOAD1,WLOAD5,WLOAD15)*. Définir ces arguments est très important pour définir les composants, puisqu'ils vont indiquer à quoi servent ces arguments.

Configuration des calendriers

Configuration des calendriers

Dans ce plugin, certaines configurations vont utiliser les calendriers pour définir :

- Période de contrôle (quand les contrôles vont être effectués : tous les jours 24/24h, uniquement les heures ouvrées de Lundi à Vendredi...)
- Période de notification (quand les notifications vont être envoyées : tous les jours 24/24h, uniquement les heures ouvrées de Lundi à Vendredi...)

Pour accéder à ce menu, il faut aller à *Plugins* > *Monitoring* > *Calendrier*

Vous créez les calendriers et définissez les jours et heures d'exécution des actions.

Configuration d'un hôte

Configuration des hôtes

Pour chaque hôte que l'on souhaite monitorer, il faut définir :

- Commande : contrôle qui vérifie que l'hôte est accessible
- Définition de contrôle
- Période de contrôle (quand les contrôles vont être effectués : tous les jours 24/24h, uniquement les heures ouvrées de Lundi à Vendredi...)

Il y a 2 possibilités pour configurer :

- Définir sur l'entité pour tous les hôtes : voir dans l'onglet *Monitoring* dans le formulaire d'entité. Tous les hôtes de cette entité vont avoir ces valeurs.
- Définir sur chaque hôte: voir dans l'onglet *Monitoring - Ressources* dans le formulaire de chaque hôte (formulaire ordinateurs, réseaux, imprimantes...).

Note: Par défaut, la configuration est faite (à l'installation ou à la mise à jour du plugin) sur l'entité racine.

Gabarits RRDTOOL

Gestion des gabarits RRDTOOL

Si on souhaite avoir/générer les graphiques RRDTOOL des ressources, les gabarits doivent être ajoutés dans le plugin. L'accès se fait via le menu *Plugins* > *Monitoring* > *Gabarits RRDTOOL*.

Les gabarits peuvent être récupérés dans le dépôt GIT *https://github.com/rrdtooltemplates/rrdtool_templates*.

Pour la génération des graphiques, 2 types de gabarits doivent être présents :

- Perfdata: utilisé pour parser les évènements et les ajouter dans un fichier RRDTOOL.
- Graph: utilisé pour générer les graphiques à partir des données issues du fichier RRDTOOL.

Gabarits de contact

Gestion des gabarits de contacts

On peut gérer les gabarits de notifications (gabarits de contact) afin de configurer comment et quand envoyer des mails quand les contrôles sont critiques.

L'accès à ce menu se fait par *Plugins* > *Monitoring* > *Gabarits de contact*.

Note: Il faut créer au moins un gabarit *par défaut*.

Après avoir créer des gabarits, il faut assigner ce gabarit à un compte utilisateur (si on ne veut pas utiliser le gabarit par défaut pour cet utilisateur). Pour faire celà, voir l'onglet *Monitoring-Contact* dans le formulaire utilisateur (menu *Administration* > *Utilisateurs*).

Configuration des composants

Configuration des composants

Ces composants sont utilisés pour définir quoi monitorer pour chaque hôte.

Pour accéder à ce menu, il faut aller à *Plugins* > *Monitoring* > *Composants*

Elements à configurer pour chaque contrôle de composant sont :

- *Nom*: donner un nom
- *Commande* : défini les commandes (voir le chapitre *configuration_pluginmonitoring_command.dita*). Ce champs est obligatoire, sauf si un contrôle à distance est défini.
- *Gabarit (pour la génération de graphiques)*: selectionner un gabarit pour la génération des graphiques (RRDTOOL).
- *Définition de contrôle* : défini le contrôle de définition pour ce composant (voir le chapitre *Configuration des définitions de contrôle* on page 21).
- *Contrôles actifs*: ce composant est-il un contrôle actif (Shinken exécute cette commande)
- *Contrôles passifs*: ce composant est un contrôle passif (c'est la machine distante qui va exécuter cette commande et envoyer à Shinken seulement lors d'un changement d'état)
- *Période pour le contrôle*: défini la période (c'est le calendrier de GLPI) qui permet de dire quand le contrôle va s'effectuer (24x7, pas les week end....)

Il est possible de définir un contrôle à distance, le serveur distant va exécuter lui-même la commande et renvoyer le résultat à Shinken). Voici les champs pour faire celà :

- *Utilitaire utilisé pour les contrôles à distance*: choisir le système à utiliser : *byssh*, *nrpe* or *nsca*.
- *Utiliser des arguments (seulement pour NRPE)*: si *nrpe* est sélectionné, il est possible de définir la commande à exécuter, donc défini ici et pas dans le fichier nrpe.cfg sur l'hôte distant. **Activer peut être dangereux pour la sécurité**
- *Commande alias si c'est requis (seulement pour NRPE)*: défini la commande and le fichier de config nrpe.cfg sur les serveurs distants.

Configuration des catalogues de composants

Configuration des catalogues de composants

Ces catalogues de composants sont utilisés pour définir pour quels hôtes le système de monitoring va contrôler les ressources.

Pour accéder à ce menu, il faut aller à *Plugins* > *Monitoring* > *Catalogue de composants*

Composants

Définir les composants de *catalogue de composants*

Dans cet onglet, il y a la liste des composants pour les hôtes de ce catalogue de composants.

Par exemple : pour un *catalogue de composants* appelé 'serveurs http', il peut avoir les composants *check_http* et *check_apachestatus*

Hôtes

Ajouter des hôtes à *catalogue de composants*

Il y a 2 méthodes pour ajouter des hôtes dans ce catalogue de composants.

Hôtes statiques

Ajouter des hôtes statiques au *catalogue de composants*

La première méthode pour ajouter des hôtes est de les ajouter manuellement par un utilisateur. Cette méthode est appelée *hôtes statiques* et est disponible dans l'onglet du même nom.

Hôtes dynamiques

Configurer les hôtes dynamiques à *catalogue de composants*

La seconde méthode pour ajouter des hôtes est de définir des règles (recherche). Cette méthode est appelée *hôtes dynamiques* et disponible dans les onglets *règles* et *hôtes dynamiques*.

Dans l'onglet *règles*, il est possible d'ajouter **seulement une règle** par type de matériel. Ces règles utilisent le moteur de recherche de GLPI. Si une règle d'ordinateur est définie par *Système d'exploitation* contient *Linux*, tous les ordinateurs qui vérifient cette recherche vont être ajoutés automatiquement dans l'onglet *hôtes dynamiques*.

Note: Quand on ajoute ou modifie un type de matériel (comme un ordinateur dans notre exemple), la règle va se rejouer et mettre à jour la liste des hôtes dynamiques sans intervention humaine. Donc quant on ajoute ou modifie un inventaire avec un outil automatique comme FusionInventory, il va mettre à jour les hôtes et des composants affectés.

Contacts

Définir les contacts de *catalogue de composants*

Dans cet onglet, il y a la liste des contact ou des groupes de contacts. C'est utilisé par Shinken pour envoyer les notifications dans le cas ou une ressource devient critique. Donc ces contacts vont être associés aux composants de hôtes configurés dans ce catalogue.

Configuration des catalogues de services

Configuration des catalogues de services

Les catalogues de services sont utilisés pour vérifier des services (mon application web est OK? Est-ce que j'ai au moins un des 3 controleurs de domaines windows qui est OK?). C'est basé sur les ressources déjà définies par les catalogues de composants. C'est plus simple d'expliquer avec un exemple.

Notre application GLPI est composée de :

- 3 serveurs Apache en load balancing
- 1 serveur MySQL

Il faut créer un catalogue de service avec 3 groupes (l'operateur entre ces groupes est : *et*):

- Groupe Apache

  On défini un opérateur logique *ou*, ce qui correspond à avoir au moins un des 3 serveurs Apache OK. On ajoute les 3 ressources :

  - *Check_http* on *serverA1*
  - *Check_http* on *serverA2*
  - *Check_http* on *serverA3*
- Groupe MySQL

  On défini un opérateur logique *ou*, mais on n'a qu'un seul serveur MySQL, donc il doit être tout le temps OK. On ajoute la ressource :

  - *Check_mysql* on *serverM1*
- Groupe temps de réponse/scénario

  On défini l'opérateur logique *ou*, mais on n'a qu'un seul contrôle sur les temps de réponse de notre application GLPI et on teste en même temps si on peut se connecter, aller sur la page de liste des ordinateurs par exemple avec JMETER.

  - Check_jmeter (attached on server you want, so it's defined on a components catalog). This check define a critical when responsive time if more than 50ms and if it can't log in GLPI application (scenario).

Maintenant que c'est configuré, on va voir les cas ou Shinken nous renvoi un état critique :

- Les 3 serveurs Apache sont arrêtés (si 2 sont arrêtés, ça restera en état OK)
- Le serveur MySQL est arrêté

- Les temps de réponse de GLPI sont plus de 50ms
- Le scénario de connexion à GLPI ne fonctionne pas

Part
IV

Utilisation quotidienne

Topics:

Utilisation quotidienne du système de monitoring

Ce chapitre décrit comment configurer les composants

Redémarrer Shinken

Redémarrer Shinken

Quant on ajoute ou modifie un équipement (configuration d'un équipement), sa configuration est modifié dans GLPI. Redémarer le module Arbiter de Shinken est indispensable pour recharger la nouvelle configuration. Celà peut être fait par un utilisateur avec la commande de redémarrage ou peut être avec un script. Nous travaillons avec l'équipe de Shinken pour avoir la possibilité de recharger cette configuration de GLPI manuellement ou automatiquement quand la configuration change.

Afficher les évènements

Afficher les évènements

Où visualiser les évènements?

Dans GLPI

Dans GLPI

Les évènements peuvent être consultés dans 2 endroits dans GLPI :

- Le dashboard général dans le menu *Plugins* > *Monitoring* > *Dashboard*. Il y a différentes vues :
    - *Services catalog*: display state of services catalogs
    - *Components catalog*: display state of components catalogs
    - *All resources*: display all resources and you can use search engine to find a resource or a type of resource.
- Dans chaque fiche de matériel, un onglet *Monitoring-resources* affiche les ressources et leur état pour cet équipement.

Si un gabarit est défini pour les composants, les graphiques pourront être affichés.

Dans l'application Android

Dans l'application Android

Affiche le nombre d'équipement dans l'état critical, warning et OK (le type de vue *Catalogue de composants* ou *Toutes les ressources* peut être défini dans le menu *options*.

Quand vous cliquez sur le boutton de couleur (vert, orange ou rouge), une nouvelle fenêtre apparait et affiche la liste des ressources/hôtes qui ont cet état.